Chapter 7
A Progressive Exposure Approach for Secure Service Discovery in Pervasive Computing Environments

S. Durga
Karunya University, India

ABSTRACT

The dynamic property of pervasive computing hinders users to have complete knowledge of the relationship among services, service providers, and credentials. The involvement of only the necessary users and service providers for service discovery in pervasive computing environments is challenging. Without prudence, users' and service providers' requests or service information, their identities, and their presence information may be sacrificed. The problem may be as difficult as a chicken-and-egg problem, in which both users and service providers want the other parties to expose sensitive information first. In this chapter, the authors propose a progressive approach to solve the problem. Users and service providers expose partial information in turn and avoid unnecessary exposure if there is any mismatch. Although 1 or 2 bits of information are exchanged in each message, the theoretical analysis and experiments show that our approach protects sensitive information with little overhead.

INTRODUCTION

In pervasive computing environments, intelligent devices are ubiquitously embedded within our personal belongings, homes, offices, and even public environments. These devices provide us various network services (services for short). Via service discovery protocols, these services are discovered just in time. Client devices and services automatically configure themselves without users' involvement. Much research has been conducted on service discovery, as reviewed in (Zhu et al., 2005). However, the problem of involving only necessary service providers and users in a service discovery session has not been well addressed. If unnecessary users and service providers are involved, then security and privacy may be sacrificed. Services may be illegally discovered or accessed and personal privacy may be exposed and inferred.

DOI: 10.4018/978-1-61520-741-1.ch007

For traditional network service accesses, it is not difficult to involve only necessary and legitimate service providers and users. Usually, a user explicitly specifies a service and supplies a credential such as a username and password pair to authenticate with a service provider. Then, the service provider verifies the user and checks the user's privilege. The user has prior knowledge of the service, service provider, credential, and relationship among them. Nevertheless, in pervasive computing environments, a user may not have such knowledge.

Challenges arise when environments change. First, a user may interact with many more services and service providers in pervasive computing environments than in conventional computing environments. For instance, a room may be saturated with hundreds of devices and services. Furthermore, everyone may become a service provider. For example, if Bob shares his MP3 player with Alice, then Bob becomes a service provider. A significant growth in the number of services and service providers makes it difficult to memorize the relationships among the services, service providers, and credentials. Second, pervasive computing environments are extremely dynamic. Devices and services may be unattended, services are added and removed, service providers' mobility causes the devices that they wear and carry to move, and partial failures cause services to be inaccessible. The dynamic property of pervasive computing hinders users to have complete knowledge of the relationship among services, service providers, and credentials.

Without such knowledge, the problem to involve only necessary service providers and users becomes difficult when users and service providers have privacy concerns. If a user is too cautious to interact with a service provider, then a user may miss the opportunity to access a service and a service provider misses an opportunity to serve a user. However, unnecessary interaction between a user and a service provider may expose a user's intent (what service a user is looking for), his credentials, and presence information. Similarly, a service provider may unnecessarily expose his service information, identity, and presence information.

Many service discovery protocols have been proposed, but it seems that no protocol addresses the problem without sacrificing security, privacy, or convenience. Several protocols and their security extensions adopt the traditional approach such that users start service discovery by supplying credentials

together with service discovery requests (Ellison, 2003, p.37) . The design is secure and only involves necessary users and service providers besides the server system. Nevertheless, both users and service providers expose their privacy to the central server system. In PrudentExposure (Zhu et al., 2006, p. 418), only users and service providers that share secrets discover and communicate with each other, but there is still a privacy leak among insiders. For example, if Bob only shares an MP3 player with Alice, then it is unnecessary to contact Bob when Alice discovers an electronic book.

We classify privacy concerns into four cases, as shown in Figure 1. The four cases are the combinations resulting from whether a user and a service provider have privacy concerns. Since there is no privacy concern in Case 1, we may directly apply authentication and authorization to secure services. In Case 2, service providers may announce their service information first because they do not have privacy concerns. Note that service providers can announce service information in an encrypted form, so only users who can decrypt messages will understand service information. If a service provider does not provide a desired service, then a user may keep silent and therefore protect the user's privacy. Similarly, in Case 3, users send their requests first. If a service provider provides the required service, and the user has privilege, then the service provider contacts the user. Otherwise, the service provider keeps silent. Nevertheless, we identify that Case 4 is as difficult as a chicken and- egg problem. That is, neither users nor service